21.06.2004

日本国特許庁 JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日 Date of Application:

2003年 6月30日

出 願 番 号 Application Number:

特願2003-187690

REC'D 0 6 AUG 2004

[ST. 10/C]:

[JP2003-187690]

WIPO PCT

出 願 人
Applicant(s):

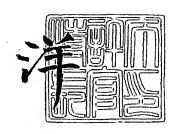
松下電器產業株式会社

PRIORITY DOCUMENT

SUBMITTED OR TRANSMITTED IN COMPLIANCE WITH RULE 17.1 (a) OR (b)

2004年 7月23日

特許庁長官 Commissioner, Japan Patent Office 1) (1)



【書類名】

特許願

【整理番号】

2968250008

【提出日】

平成15年 6月30日

【あて先】

特許庁長官 殿

【国際特許分類】

G05B 9/06

【発明者】

【住所又は居所】

東広島市鏡山3丁目10番18号 株式会社松下電器情

報システム広島研究所内

【氏名】

今西 祐子

【発明者】

【住所又は居所】

東広島市鏡山3丁目10番18号 株式会社松下電器情

報システム広島研究所内

【氏名】

土井 繁則

【特許出願人】

【識別番号】

000005821

【氏名又は名称】 松下電器産業株式会社

【代理人】

【識別番号】

100090446

【弁理士】

【氏名又は名称】

中島 司朗

【手数料の表示】

【予納台帳番号】

014823

【納付金額】

21,000円

【提出物件の目録】

【物件名】

明細書

【物件名】

図面 1

【物件名】

要約書 1

9003742

【包括委任状番号】

要

【プルーフの要否】



【書類名】 明細書

【発明の名称】 ガーベジコレクションシステム

【特許請求の範囲】

【請求項1】 複数スレッドで構成されるオブジェクト指向プログラムの実 行過程において不要になったオブジェクトに対応するメモリ領域を解放するガー ベジコレクションシステムであって、

複数スレッドそれぞれを順に選択する選択手段と、

選択されたスレッドについて、当該スレッドの実行を停止して、当該スレッドからオブジェクトポインタの参照を通じてアクセス可能であるオブジェクトを検出して当該検出したオブジェクトを非解放対象として管理して、当該スレッドの実行を再開するという手順からなる検査処理を実施する検査手段と、

前記選択手段による前記選択が開始された後において、実行中のスレッドによりオブジェクトポインタが処理対象にされたことを検知した場合には、当該オブジェクトポインタが指すオブジェクトを非解放対象として管理する検知手段と、

前記複数スレッド全てについて前記検査処理が完了した後において、非解放対象として管理されているオブジェクト以外のオブジェクトに対応するメモリ領域を、解放する解放手段とを備える

ことを特徴とするガーベジコレクションシステム。

【請求項2】 前記検知手段は、前記検知を、実行中のスレッドについて未 だ検査処理がなされていない場合に限り行い、

前記検知手段は、

前記検知をした場合に、当該実行中のスレッドに処理対象にされたオブジェクトポインタ、及び当該オブジェクトポインタから辿ることのできるオブジェクト内のオブジェクトポインタを、当該スレッドに対応する作業用メモリ領域に保存する検出部と、

前記検査手段によりスレッドの実行が停止されている間に、当該スレッドに対応する作業用メモリ領域内のオブジェクトポインタから辿ることのできるオブジェクトを非解放対象として管理する管理部とを有する

ことを特徴とする請求項1記載のガーベジコレクションシステム。



【請求項3】 前記検査処理は、

選択された当該スレッドに対応するスタック内のオブジェクトポインタが指す オブジェクトを、前記アクセス可能であるとして検出した際において、

検出した当該オブジェクトが既に非解放対象として管理されておらず、かつ、 当該オブジェクト内にオブジェクトポインタがあるときに限り、当該オブジェクトポインタが指すオブジェクトを更に前記アクセス可能であるとして検出する手順を、

繰り返し行うことを内容とする処理であり、

前記選択手段は、最初の選択後は、前記検査手段により前記検査処理が行われた後において前記複数スレッドのうち前記検査処理を実施されていないスレッドがある限り更なる選択を行い、

前記選択手段は、各スレッドに関する情報を参照し、所定のスレッド選定条件に基づき、前記選択を行う

ことを特徴とする請求項2記載のガーベジコレクションシステム。

【請求項4】 前記スレッド選定条件は、スレッド状態がwait状態であるスレッドをスレッド状態がwait状態以外であるスレッドよりも早期に選択することを示す条件を含み、

前記選択手段は、前記選択を行う時点でwait状態のスレッドがある限り、 当該wait状態のスレッドを選択する

ことを特徴とする請求項3記載のガーベジコレクションシステム。

【請求項5】 前記スレッド選定条件は、スレッド優先度のスレッド優先度の低いスレッドをスレッド優先度の高いスレッドよりも早期に選択することを示す条件を含む

ことを特徴とする請求項3又は請求項4記載のガーベジコレクションシステム

【請求項6】 前記スレッド選定条件は、スレッドに対応するスタックサイズが小さいスレッドをスタックサイズが大きいスレッドよりも早期に選択することを示す条件を含む

ことを特徴とする請求項3~5のいずれか1項に記載のガーベジコレクション



システム。

【請求項7】 コンピュータ上において複数スレッドで構成されるオブジェクト指向プログラムの実行過程において不要になったオブジェクトに対応するメモリ領域を解放するガーベジコレクション方法であって、

複数スレッドそれぞれを順に選択する選択ステップと、

選択されたスレッドについて、当該スレッドの実行を停止して、当該スレッドからオブジェクトポインタの参照を通じてアクセス可能であるオブジェクトを検出して当該検出したオブジェクトを非解放対象として管理して、当該スレッドの実行を再開するという手順からなる検査処理を実施する検査ステップと、

前記選択ステップによる前記選択が開始された後において、実行中のスレッドによりオブジェクトポインタが処理対象にされたことを検知した場合には、当該オブジェクトポインタが指すオブジェクトを非解放対象として管理する検知ステップと、

前記複数スレッド全てについて前記検査処理が完了した後において、非解放対象として管理されているオブジェクト以外のオブジェクトに対応するメモリ領域を、解放する解放ステップとを含む

ことを特徴とするガーベジコレクション方法。

【請求項8】 複数スレッドで構成されるオブジェクト指向プログラムの実行過程において不要になったオブジェクトに対応するメモリ領域を解放するガーベジコレクション処理をコンピュータに実行させるためのコンピュータプログラムであって、

前記ガーベジコレクション処理は、

複数スレッドそれぞれを順に選択する選択ステップと、

選択されたスレッドについて、当該スレッドの実行を停止して、当該スレッドからオブジェクトポインタの参照を通じてアクセス可能であるオブジェクトを検出して当該検出したオブジェクトを非解放対象として管理して、当該スレッドの実行を再開するという手順からなる検査処理を実施する検査ステップと、

前記選択ステップによる前記選択が開始された後において、実行中のスレッド によりオブジェクトポインタが処理対象にされたことを検知した場合には、当該



オブジェクトポインタが指すオブジェクトを非解放対象として管理する検知ステップと、

前記複数スレッド全てについて前記検査処理が完了した後において、非解放対象として管理されているオブジェクト以外のオブジェクトに対応するメモリ領域を、解放する解放ステップとを含む

ことを特徴とするコンピュータプログラム。

【発明の詳細な説明】

[0001]

【発明の属する技術分野】

本発明は、アプリケーションプログラム(AP:Application Program)が使用したメモリについてのガーベジコレクション(GC:Garbage Collection)に関する。

[0002]

【従来の技術】

従来のオブジェクト指向のプログラミング言語には、APの作成者にメモリ領域の確保や解放を意識させず、APが使用した後に不要となったオブジェクト(オブジェクトインスタンス)に対応するメモリ領域の解放を実行環境側に任せる方法を採るものがある。例えばJava(登録商標)言語である。なお、Java(登録商標)は米国Sun Microsystems社の商標である。

[0003]

このような言語を用いて記述されたAPは、不要となったオブジェクトに対応 するメモリ領域を自動的に解放するガーベジコレクション(GC)機構を備える 実行環境上で動かされる。

GC機構は、APの実行に際して動的に確保されたオブジェクトに対応するメモリ領域がどこからも参照されていない状態になっていることを検出してそのメモリ領域を解放し再利用可能状態にする。

[0004]

マルチスレッド構成のAPにおいて、各スレッドは、それぞれスタック領域と 対応し、動作過程において、スタック領域内にデータを格納したり、格納したデ



ータを参照したり、オブジェクトを生成したりする。オブジェクトを生成した場合には通常はスタック領域内にそのオブジェクトのポインタ、つまりそのオブジェクトのメモリ内における所在位置を指し示すデータ(以下、「オブジェクトポインタ」という。)が格納され、スレッドからのオブジェクトへのアクセスはそのオブジェクトポインタを参照することによりなされる。また、通常、オブジェクト内領域にも、他のオブジェクトへのオブジェクトポインタが格納される。

[0005]

ある時点においてAPにより参照されている全てのオブジェクトは、APの各スレッドに対応する各スタック領域内に含まれるオブジェクトポインタから、直接、又は1以上のオブジェクト内のオブジェクトポインタを媒介にして、辿ることができる。

これに対応してGC機構は、基本的に、ある時点においてスタック領域内のオブジェクトポインタから辿れなくなっているオブジェクトに対応するメモリ領域を不要なものとして解放対象にする。

[0006]

従来のGC方式として知られているマークアンドスイープ方式は、特定のオブジェクトポインタから辿れる全てのオブジェクトにマークを付けていく処理を行った後に、全てのオブジェクトを走査して、マークの付いていないオブジェクトに対応するメモリ領域を解放する方式である。

このマークアンドスイープ方式のGCをマルチスレッド構成のAPの実行時に 行う場合に、GCの迅速性を最優先して単純に、全スレッドを停止してから、各 スレッドに対応するスタック領域内のオブジェクトポインタから辿れる全てのオ ブジェクトにマークを付与する処理を行い、その後に全スレッドの停止を解除し て、マークの付いていないオブジェクトの解放を行うとするならば、次の問題が 生じる。

[0007]

即ち、APの全スレッドが停止している期間が長期化する可能性があり、この場合に、ユーザの操作等に対して一切反応を示せず、例えばコンピュータのディスプレイの表示内容は変化しないままとなり、ユーザを戸惑わせてしまうという



問題である。

この問題を解決する方式として、このマークアンドスイープを用いたGCを、マルチスレッド構成のAPを全く停止することなく実行する方式が提案されている(特許文献1参照)。

[0008]

この方式は、ルートノードなるオブジェクトから辿れる全てのオブジェクト及びスレッド毎の各スタック領域内のオブジェクトポインタから辿れる全てのオブジェクトに対してマークを付与する第1処理を行い、その第1処理中に、APのスレッド(以下、「APスレッド」という。)の動作によりオブジェクトへのオブジェクトポインタが移動した場合にそのオブジェクトを表すデータをマークスタックなる領域に積んでおき、そのマークを付与する処理が完了した段階で、更にマークスタックから辿れる全てのオブジェクトに対してマークを付与する第2処理を行い、最後に、マークされていないオブジェクトに対応するメモリ領域を解放するものである。

[0009]

【特許文献1】

特許第3027845号公報

[0010]

【発明が解決しようとする課題】

しかしながら、上述のAPを全く停止せずにマークを付与する方式では、AP スレッドの動作によってスタック領域内のデータが変化するため、第1処理中、 スタック領域内のオブジェクトポインタから辿れるオブジェクトに対してマーク を付与する処理の一部が無駄となる可能性がある。

[0011]

例えば、GCを行うスレッドが、APの1つのスタック領域内のオブジェクトポインタ(オブジェクトポインタA)を検出して、そのオブジェクトポインタAから辿れるオブジェクトに対してマークを付与する処理Aを実行している間に、そのスタック領域に対応するAPスレッドが、オブジェクトポインタA或いはそれらのオブジェクト内の1又は複数のオブジェクトポインタを、コピーしてスタ



ック領域に新たに格納する動作をしたならば、GCを行うスレッドはその処理Aの終了後にいずれ、再び処理Aと一部重複する処理を行う、或いはその重複防止のためのチェック処理をわざわざ行うことになり、これが無駄となる。この処理の無駄は、GCの開始から完了までに要するCPU時間を無駄に増加させることにつながり、CPUの利用効率を低下させる。

[0012]

そこで、本発明は、上記問題に鑑みてなされたものであり、APの全スレッドが停止している期間を長期化させないようにするとともに、GCの開始から完了までに要するCPU時間の無駄な増加をある程度抑制するようなGC方式を用いるガーベジコレクション(GC)システムを提供することを目的とする。

[0013]

【課題を解決するための手段】

上記課題を解決するために、本発明に係るGCシステムは、複数スレッドで構成されるオブジェクト指向プログラムの実行過程において不要になったオブジェクトに対応するメモリ領域を解放するガーベジコレクションシステムであって、複数スレッドそれぞれを順に選択する選択手段と、選択されたスレッドについて、当該スレッドの実行を停止して、当該スレッドからオブジェクトポインタの参照を通じてアクセス可能であるオブジェクトを検出して当該検出したオブジェクトを非解放対象として管理して、当該スレッドの実行を再開するという手順からなる検査処理を実施する検査手段と、前記選択手段による前記選択が開始された後において、実行中のスレッドによりオブジェクトポインタが処理対象にされたことを検知した場合には、当該オブジェクトポインタが指すオブジェクトを非解放対象として管理する検知手段と、前記複数スレッド全でについて前記検査処理が完了した後において、非解放対象として管理されているオブジェクト以外のオブジェクトに対応するメモリ領域を、解放する解放手段とを備えることを特徴とする。

[0014]

上記構成により、APスレッドからスタック内やオブジェクト内のオブジェクトポインタを介して辿ることができるオブジェクトについての非解放対象として



の指定の過程つまりマーク付与の過程においては、そのAPスレッドを停止させているため、そのAPスレッドが動作することに起因してスタック領域内のデータが変化するという事態が起こらないため、その過程におけるマーク付与が無駄になる危険性はなくなり、この点についてはCPUの利用効率低下を防止できる。また、これにより、全てのAPスレッドを停止して上述のマーク付与を行っているのではないため、APの全スレッドが停止している期間の長期化が防止できる。

[0015]

【発明の実施の形態】

<実施形態1>

以下、本発明の実施形態1に係るガーベジコレクション(GC)システムについて図面を用いて説明する。

<構成>

図1は、本発明の実施形態1に係るGCシステムの機能ブロック図である。

[0016]

GCシステム10は、CPU、メモリ等を備えたコンピュータにおいてメモリ に格納された制御プログラムがCPUによって実行されることにより実現され、マルチスレッド制御を行う一般のオペレーティングシステム(OS:Operating System)及びいわゆる仮想マシンを含み、Java(登録商標)言語等で作成されたアプリケーションプログラムの実行環境として位置づけられるシステムである。

[0017]

GCシステム10は、図1に示すように、インタプリタ部100、オブジェクト管理部200、スレッド管理部300及びGC部400を備える。

ここで、インタプリタ部100は、基本的にインタプリタであってAPを実行する機能を担い、命令実行部110及びオブジェクト参照検出部120を有する

[0018]

オブジェクト管理部200は、オブジェクトを管理する機能を担い、from



テーブル221、toテーブル222、オブジェクト等に対応するメモリ領域であるオブジェクト情報記憶部210、オブジェクト生成部230及びテーブル切替部240を有する。

なお、fromテーブル 221 は、GC 開始前において存在する全オブジェクトに対する全オブジェクトポインタを格納するものと位置づけられるテーブルであり、to テーブル 222 は、マークアンドスイープ方式でのマーク付与がなされたオブジェクトに対するオブジェクトポインタを格納するために利用されるテーブルである。これらのテーブルについては後に詳細に説明する。

[0019]

スレッド管理部300は、マルチスレッド制御を実現し、スレッドを管理する機能を担うものであり、スレッド制御部310と、スレッド毎に、スレッド情報320、オブジェクト参照情報330及びスタック340に対応するメモリ領域を有する。

なお、スレッド情報320は、GC処理全体の開始時点でONにされそのスレッドに対しての参照処理が終了するとOFFにされるGCフラグを含む。参照処理は、オブジェクトへのマーク付与に相当する処理であり、オブジェクトを参照するためにスタック内等に格納されているオブジェクトポインタと同一内容のオブジェクトポインタをfromテーブルからtomテーブルに移動することを主内容とする処理である。

[0020]

スレッド管理部300は、各スレッドを並列的に実行する。即ち、スレッド管理部300は、例えば微小時間毎に各スレッドを切り替えて実行する、或いは各スレッドをマルチプロセッサにおける複数のプロセッサエレメントによって並列実行する。

GC部400は、基本的にいわゆるガーベジコレクタに相当し、GC制御部410、スレッド選定条件記憶部420、参照処理部430及び解放部440を有する。このGC部400を中心として行われるGCは、基本的にマークアンドスイープ方式を用い、参照処理の対象となるAPスレッドを順番に選択して、選択したスレッドを停止させて参照処理を行ってからその停止を解除する方式のもの



である。

[0021]

インタプリタ部100における命令実行部110は、APスレッドを構成する 命令列を逐次解釈して実行し、命令がオブジェクトの生成命令である場合にはオ ブジェクト生成部230にオブジェクトを生成させる機能を有する。

オブジェクト参照検出部120は、GCフラグがONである場合に限って、命令実行部110により実行されるスレッド内の命令が、オブジェクトのオブジェクトポインタを処理対象とする命令であるときに、そのオブジェクトポインタを、オブジェクト参照情報330のメモリ領域の中に格納する機能を有する。

[0022]

オブジェクト管理部200におけるオブジェクト生成部230は、いわゆるクラスファイル等のオブジェクトの定義情報を参照して、OSのメモリ管理機構によりオブジェクトに対応するメモリ領域を確保することにより、メモリ内にオブジェクトを生成する機能を有する。

テーブル切替部 240は、fromテーブルを指すものとして用いられているポインタとtoテーブルを指すものとして用いられているポインタとを交換することにより、瞬時にfromテーブルの内容とtoテーブルの内容とを交換したに等しい効果を生じさせる機能を有する。

[0023]

スレッド管理部300におけるスレッド制御部310は、マルチスレッド制御を行い、APを構成する各スレッド及びGC処理のスレッドを擬似並行的に実行する。

GC部400におけるスレッド選定条件記憶部420は、参照処理を行う対象となるスレッドを選択するための条件を示すスレッド選定条件を記憶しているメモリ領域である。

[0024]

参照処理部430は、参照処理を行う機能を有し、解放部440は、マーク付与がなされていないオブジェクトの解放、即ち、GCの終了段階においてfromテーブル221に残存しているオブジェクトポインタが指すオブジェクトに対



応するメモリ領域を解放する機能を有する。

また、GC制御部410は、GC制御処理を実行する機能を有する。即ち、GC制御部410は、スレッド選定条件を参照することにより、参照処理の対象にするAPスレッドを1つ選定し、選択したスレッドについて、スレッド制御部310にそのスレッドを停止させてから、参照処理部430にそのスレッドに対応するスタック及びオブジェクト参照情報に基づいて参照処理を行わせ、その後にスレッド制御部310に選択したスレッドの停止を解除させてから、次のAPスレッドの選択を行うという手順を、未処理のAPスレッドが存在しなくなるまで繰り返してから、解放部440に、マーク付与がなされていないオブジェクトの解放を行わせる機能を有する。

[0025]

くデータ>

以下、GCシステム10において取り扱われるデータについて説明する。

図2は、オブジェクトとオブジェクトポインタとの関係を例示する図である。 オブジェクトを指すつまりオブジェクトの所在位置を指すオブジェクトポイン タは、オブジェクト内、スタック内、又は共有オブジェクト管理情報353内に 存在し得る。ここで、共有オブジェクト管理情報353は、APの実行環境とし ての仮想マシンにおいて必要なため確保されているオブジェクト群に対するオブ ジェクトポインタ群を含むデータである。

[0026]

図2の例では、共有オブジェクト管理情報353内のあるオブジェクトポインタがオブジェクト201cを指している。

また、スレッド351に対応するスタック内のあるオブジェクトポインタは、オブジェクト201aを指しており、つまり、オブジェクト201aのメモリアドレスを内容としており、また別のオブジェクトポインタは、オブジェクト201dを指している。従って、スレッド351は、実行中にこれらのオブジェクトポインタを参照することによりオブジェクト201a、オブジェクト201dにアクセス可能な状態である。

[0027]



また、スレッド352に対応するスタック内のあるオブジェクトポインタは、 オブジェクト201bを指しており、オブジェクト201b内のデータとして、 オブジェクト201eを指すオブジェクトポインタが含まれている。従って、ス レッド352は、これらのオブジェクトポインタを辿ることによりオブジェクト 201eにアクセス可能な状態である。

[0028]

図3は、fromテーブル及びtoテーブルを示す図である。

オブジェクト情報記憶部210内に、オブジェクトポインタ又はnull値を 十分な数だけ格納するための2つのテーブルが設けられており、また、from テーブルポインタ211及びtoテーブルポインタ212が存在する。from テーブルポインタ211は、その2つのテーブルのうち一方を指すポインタであ り、toテーブルポインタ212は、他方を指すポインタである。ここでは、現 時点でfromテーブルポインタ211が指しているテーブルをfromテーブ ルと称しており、toテーブルポインタ212が指しているテーブルをtoテー ブルと称している。

[0029]

fromテーブル221は、GC開始時点においては、その時に存在する全オブジェクトに対する全オブジェクトポインタを格納している。

[0030]

図4は、参照処理によるfromテーブル及びtoテーブルの内容の変化を示す図であり、図4(a)は参照処理開始前の状態を示し、図4(b)は参照処理後の状態を示す。

図4 (a) では、参照処理開始前におけるfromテーブル221xは、objA202a、objB202b、objC202cのそれぞれのオブジェクト



を指すオブジェクトポインタを含んでいる。

[0031]

[0032]

図5は、スレッド情報、スタック及びオブジェクト参照情報を示す図である。 スレッド情報320は、スレッド毎に、スレッド生成時に生成され、そのスレッドについての情報を含むものであり、具体的には、状態321、優先度322、スタックポインタ323、GC開始時スタックポインタ324、GCフラグ325、オブジェクト参照情報先頭ポインタ326及びオブジェクト参照情報カレントポインタ327を含む。なお、スレッド生成時には、スレッド情報320のメモリ領域のみならずスタック340のメモリ領域も確保される。

[0033]

スレッド情報320における状態321は、マルチスレッド制御のためのスレッド状態を示す情報であり、wait状態、run状態、ready状態等の別を示す。

優先度322は、スレッドの優先度を示す情報であり、優先度は例えばスレッド生成時にAPからの指定を受けて定まる。なお、マルチスレッド制御において優先度が高いスレッドがready状態であればそれより優先度の低いスレッドよりも優先的に実行対象とされる。

[0034]

スタックポインタ323は、該当のスレッドについてのスタック内における現在の有効なデータ範囲の終端を示すものである。なお、マルチスレッド制御にお



いて、スレッドが r u n 状態から、他の状態つまり停止した状態に切り替えられる際に、それまでスタックポインタを指す所定レジスタに格納されていた値がこのスタックポインタ 3 2 3 に格納され、スレッドが r u n 状態に切り替えられる際に、このスタックポインタ 3 2 3 内の値がその所定レジスタに設定される。

[0035]

GC開始時スタックポインタ324は、GC開始時におけるスタック内における有効なデータ範囲の終端を示すものである。

GCフラグ325は、GC処理全体の開始時点でONにされ、そのスレッドに対しての参照処理が終了するとOFFにされる。

オブジェクト参照情報先頭ポインタ326は、該当スレッドに対応するオブジェクト参照情報のメモリ領域の先頭を示すものであり、スレッド生成の際のそのメモリ領域確保時に設定される。

[0036]

また、オブジェクト参照情報カレントポインタ327は、該当スレッドに対応 するオブジェクト参照情報のメモリ領域において、オブジェクト参照検出部12 0が次にオブジェクトポインタを格納すべき位置を示す情報であり、オブジェクト参照検出部120よって参照及び更新される。

図6は、参照処理とAPスレッドの状態との関係を示す図である。

[0037]

GCスレッド356は、GC制御部410によるGCを実行するスレッドであり、GCとして各APスレッドを順番に対象として参照処理を行う。ここでは、APスレッド355a、APスレッド355cの順に参照処理が行われる場合の例を示している。

APスレッド355aは、参照処理がなされた後の状態であり、実行中である。ここで実行中とは、sleep状態つまり停止状態ではないことをいい、各瞬時においてはrun状態やready状態等と変化し得る。

[0038]

APスレッド355bは、参照処理がなされているところの状態でありスレッドは停止している。参照処理は、APスレッドを強制的に停止状態にして、スタ



ック及びオブジェクト参照情報を参照して行われる。

APスレッド355cは、参照処理が未だなされていない状態であり、実行中である。

[0039]

図7は、スレッド選定条件の内容を示す図である。

スレッド選定条件421は、どのスレッドを優先して参照処理の対象とするかの判断基準となる情報であり、スレッド状態422、スレッド優先度423及びスタックサイズ424から構成される。

スレッド状態 4 2 2 は、どの状態のスレッドを優先するかを示す情報であり、 図 7 の例はwait状態のスレッドを、run状態等のスレッドより優先する旨 を示している。

[0040]

スレッド優先度423は、スレッドについて定められている優先度の高いものを参照処理の対象として優先的に選定するか、又は低いものを優先的に選定するかを示す情報であり、図7の例では優先度の低いものを優先的に選定する旨を示している。

また、スタックサイズ424は、スレッドに対応するスタック領域の有効範囲の大きさが、大きいところのスレッドを優先的に選定するか、又は小さいところのスレッドを優先的に選定するかを示す情報であり、図7の例ではスタック領域の有効範囲の大きさが小さいものを優先的に選定する旨を示している。

[0041]

<動作>

以下、GCシステム10の動作について説明する。

図8は、GC制御処理を示すフローチャートである。

- GC制御処理は、タイマーに基づき一定周期で行われる。
- GC制御部410は、最初に、fromテーブルポインタ211とtoテーブルポインタ212の内容を交換することにより、fromテーブルとtoテーブルとの内容を切り替える(ステップS11)。これにより、現在解放されていない全てのオブジェクトを指す全てのオブジェクトポインタがfromテーブルに



格納されている状態になる。

[0042]

続いて、GC制御部410は、スレッド制御部310にAPの全スレッドを停止させ(ステップS12)、APの全スレッドについてのスレッド情報中のGCフラグ325をONにし、現在のスタックポインタをスレッド情報中のGC開始時スタックポインタ324に設定し(ステップS13)、スレッド制御部310にAPの全スレッドの停止を解除させる(ステップS14)。ステップS13においては、更に、APのスレッド毎に、オブジェクト参照情報のメモリ領域を確保し、そのメモリ領域の先頭を指すポインタをオブジェクト参照情報先頭ポインタ326及びオブジェクト参照情報カレントポインタ327に設定する。

[0043]

なお、スレッド制御部310は、従来のOSのマルチスレッド制御機構と同様の方式によりスレッドの停止及び停止解除を行う。このスレッドの停止は、スレッドを停止状態、即ちsleep状態にする制御であり、スレッドの停止解除とは、sleep状態を解除して、そのsleepの前の状態に戻す制御である。

ステップS14に続いて、GC制御部410は、共有オブジェクト管理情報353内のオブジェクトポインタを辿ることで到達するオブジェクトに対するマーク付与を行うための共有オブジェクト参照処理を行い(ステップS15)、GC未処理スレッドつまり未だ参照処理の対象として選定されていないスレッドが存在するか否かを判定する(ステップS16)。なお、共有オブジェクト参照処理については後述する。

[0044]

ステップS16においてGC未処理スレッドが存在すると判定した場合には、GC制御部410は、参照処理の対象となるスレッドを決定するための対象スレッド決定処理を行い(ステップS17)、スレッド制御部310にその決定したスレッドを停止させ(ステップS18)、その決定したスレッドに関しての参照処理を内容とする対象スレッド参照処理を実行し(ステップS19)、その決定したスレッドのGCフラグ325をOFFにし(ステップS20)、スレッド制御部310にその決定したスレッドの停止を解除させて(ステップS21)、再



度、ステップS16の判定に戻る。なお、対象スレッド決定処理及び対象スレッド参照処理については、後述する。また、ステップS20の直後に、GC制御部410は、対象スレッドに対応するオブジェクト参照情報のメモリ領域を解放する。

[0045]

また、ステップS16において、GC未処理スレッドが存在しないと判定した場合には、GC制御部410は、解放部440にマーク付与がなされていないオブジェクトを解放させ(ステップS22)、GC制御処理を終える。ステップS22においては、解放部440は、参照処理によってtoテーブルにオブジェクトポインタが移行されていないところのオブジェクト、つまりfromテーブルに残存しているオブジェクトポインタが指すオブジェクトに対応するメモリ領域を解放する。この解放は、オブジェクト生成部230が、オブジェクト生成のためにオブジェクトに割り当てるためのメモリ領域を、一般的なOSのメモリ管理機能に基づき確保することに対応し、その確保されたメモリ領域の解放を意味する。

[0046]

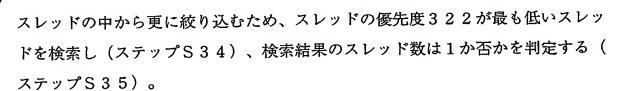
図9は、対象スレッド決定処理を示すフローチャートである。

GC制御部410は、スレッド選定条件記憶部420内のスレッド選定条件4 21を参照して、対象スレッド決定処理を行う。

まず、G C制御部410は、A Pの各スレッドに対応するスレッド情報を参照して、状態321がwait状態のスレッドを検索する(ステップS31)。この検索結果のスレッド数を判定し(ステップS32)、1であれば、その検索結果のスレッドを参照処理の対象スレッドとして決定し(ステップS37)、対象スレッド決定処理を終える。

[0047]

また、ステップS32において検索結果のスレッド数が0であれば、GC制御部410は、スレッドの優先度322が最も低いスレッドを検索し(ステップS33)、検索結果のスレッド数は1か否かを判定する(ステップS35)。また、ステップS32において検索結果のスレッド数が2以上であれば、検索結果の



[0048]

ステップS35においてスレッド数が1と判定した場合には、GC制御部410は、その検索結果のスレッドを参照処理の対象スレッドとして決定し(ステップS37)、対象スレッド決定処理を終える。

また、ステップS35において、検索結果のスレッド数が1でないと判定した場合、即ち、最も低い優先度のスレッドが複数存在した場合には、GC制御部410は、検索結果のスレッドの中からスタックサイズが最小のスレッド検索し(ステップS36)、スタックサイズが最小のスレッドを参照処理の対象スレッドとして決定し(ステップS37)、対象スレッド決定処理を終える。

[0049]

図10は、共有オブジェクト参照処理を示すフローチャートである。

GC制御部410は、オブジェクトポインタ群を含む共有オブジェクト管理情報353の先頭のオブジェクトポインタに着目して(ステップS41)、参照処理部430に参照処理(ステップS42)を行わせることにより、APの実行環境としての仮想マシンにおいて必要なため確保されているオブジェクト群全てにマーク付与を行う。

[0050]

図11は、対象スレッド参照処理を示すフローチャートである。

GC制御部410は、対象スレッドに対応するスレッド情報中のGC開始時スタックポインタ324が指す位置に着目し(ステップS51)、参照処理部430に参照処理(ステップS52)を行わせることにより、スタック領域内のオブジェクトポインタから辿ることのできる全てのオブジェクトへのマーク付与を行う。なお、GC制御処理(図8参照)の開始時点と、この対象スレッド参照処理を開始する時点は異なるため、GC開始時スタックポインタ324が指す位置とそれに続く各位置の内容は、GC開始時点とは異なっているが、後続するステップS53及びステップS54によって、GC開始以後に、対象スレッドの動作に



よって変化したオブジェクトポインタから辿ることのできる全てのオブジェクトへのマーク付与が行われるので、過剰にマーク付与をする場合はあり得るが、参照されているにも関わらずマーク付与が漏れるという事態は生じない。

[0051]

続いて、GC制御部410は、オブジェクト参照情報先頭ポインタ326の指す位置に着目し(ステップS53)、参照処理部430に参照処理(ステップS54)を行わせ、対象スレッド参照処理を終える。

図12は、参照処理を示すフローチャートである。

参照処理部430は、着目されているメモリ位置から、オブジェクトポインタを検索し(ステップS61)、オブジェクトポインタを検出したか否かを判定し(ステップS62)、検出した場合にはその検出したオブジェクトポインタと同値のオブジェクトポインタがfromテーブルに存在するか否かを判定し(ステップS64)、fromテーブルに存在すれば、fromテーブルからtoテーブルにそのオブジェクトポインタをコピーして、fromテーブル内のそのオブジェクトポインタが存在していた位置にnull値を記録し(ステップS66)、そのオブジェクトポインタが指すオブジェクトのデータ先頭に着目し(ステップS67)、再度ステップS61に戻る。

[0052]

また、ステップS64において、fromテーブルにはそのオブジェクトポインタが存在しないと判定した場合には、既にfromテーブルからtoテーブルに移されているため、参照処理部430は、着目している位置の次の位置に着目し(ステップS65)、再度ステップS61に戻る。ステップS65及びステップS61の組み合わせにより、共有オブジェクト管理情報353内のオブジェクトポインタを次々と検索することや、スタック内のオブジェクトポインタを次々と検索することや、オブジェクト参照情報内のオブジェクトポインタを次々と検索することや、あるオブジェクト内のデータメンバとしてのオブジェクトポインタを次々と検索すること等が実現される。

[0053]

また、ステップS62においてオブジェクトポインタを検出できなかったと判



定した場合には、参照処理部430は、着目位置がオブジェクト内であるか否かを判定する(ステップS63)。なお、オブジェクトポインタを検出できなかったというのは、着目位置が共有オブジェクト管理情報353内であればその共有オブジェクト管理情報353内においてオブジェクトポインタを検出できなかったということであり、着目位置がスタック内であればスタック内においてオブジェクトポインタを検出できなかったということであり、着目位置がオブジェクト参照情報内であればそのオブジェクト参照情報内においてオブジェクトポインタを検出できなかったということであり、着目位置がオブジェクトポインタを検出できなかったということであり、着目位置がオブジェクト内であればそのオブジェクト内においてオブジェクトポインタを検出できなかったということである。

[0.054]

ステップS63において着目位置がオブジェクト内であると判定した場合には、参照処理部430は、そのオブジェクト内に着目前の着目位置の次位置に着目し(ステップS68)、再度ステップS61に戻る。このステップS68によって、参照処理部430は、ステップS67によってオブジェクト内に着目する前に、着目していたオブジェクトポインタの次の位置に着目することになる。

[0055]

また、ステップS63において着目位置がオブジェクト内ではないと判定した場合には、参照処理部430は、参照処理を終える。

図13は、命令実行処理を示すフローチャートである。

インタプリタ部100における命令実行部110は、run状態にされたAPのスレッドにおけるプログラム中の命令記述を逐次解釈して実行する命令実行処理を行う。

[0056]

まず、命令実行部 1 1 0 は、スレッドの現在の実行位置における命令を解釈して実行し(ステップ S 7 1)、オブジェクト参照検出部 1 2 0 は、そのスレッドに対応するスレッド情報中のGCフラグ 3 2 5 がONであるか否かを判定し(ステップ S 7 2)、ONでなければ、命令実行部 1 1 0 が次の命令の解釈実行を行う(ステップ S 7 1)。



[0057]

ステップS72において、GCフラグ325がONであると判定した場合には、オブジェクト参照検出部120は、ステップS71において命令実行部110により実行された命令が、オブジェクトポインタを処理対象とする命令であったか否かを判定し(ステップS73)、オブジェクトポインタを処理対象とする命令でなかったならば、命令実行部110が次の命令の解釈実行を行う(ステップS71)。なお、オブジェクトポインタを処理対象とする命令とは、スタック内のオブジェクトポインタをオブジェクト内にコピーする演算や、オブジェクト内のオブジェクトポインタを他のオブジェクト内にコピーする演算である。

[0058]

ステップS 7 3 において、オブジェクトポインタを処理対象とする命令であったと判定した場合には、オブジェクト参照検出部 1 2 0 はそのオブジェクトポインタと同値のものがオブジェクト参照情報領域に既に格納済みであるか否かを判定し(ステップS 7 4)、格納済みである場合には、命令実行部 1 1 0 が次の命令の解釈実行を行う(ステップS 7 1)。

[0059]

ステップS74において、そのオブジェクトポインタと同値のものがオブジェクト参照情報のメモリ領域に格納済みでない場合には、オブジェクト参照検出部120は、そのオブジェクトポインタを、オブジェクト参照情報カレントポインタ327が指すオブジェクト参照情報中の位置に格納し(ステップS75)、そのオブジェクトポインタの指すオブジェクトに着目して、オブジェクトチェーン追跡処理(ステップS76)を行い、その後は命令実行部110が次の命令の解釈実行を行う(ステップS71)。

[0060]

なお、ステップS 7 1 における実行対象とされる命令がオブジェクトを生成する命令である場合には、命令実行部 1 1 0 は、オブジェクト生成部 2 3 0 にオブジェクトの生成を指示し、これを受けてオブジェクト生成部 2 3 0 は、オブジェクトを生成するとともにそのオブジェクトを指すオブジェクトポインタをスレッドに対して返却し、そのオブジェクトポインタのコピーを t o テーブル内に設定



する。

[0061]

図14は、オブジェクトチェーン追跡処理を示すフローチャートである。

オブジェクト参照検出部120は、着目しているオブジェクトに未着目のオブジェクトポインタが存在するか否かを判定し(ステップS81)、存在する場合には、その未着目の1つのオブジェクトポインタに着目し(ステップS82)、その着目したオブジェクトポインタをオブジェクト参照情報のメモリ領域に格納し(ステップS83)、その着目したオブジェクトポインタの指すオブジェクトに着目して更にオブジェクトチェーン処理(ステップS81~S84)を行う(ステップS84)。なお、オブジェクトポインタをオブジェクト参照情報のメモリ領域に格納したときには、オブジェクト参照検出部120は、オブジェクト参照情報カレントポインタをそのオブジェクトポインタのサイズ分だけ進める。

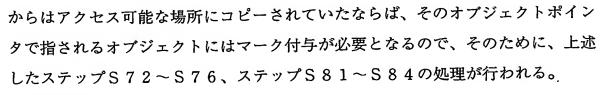
[0062]

また、ステップS81において、着目しているオブジェクトに未着目のオブジェクトポインタが存在していないと判定した場合には、オブジェクト参照検出部120は、オブジェクトチェーン追跡処理を終える。

従って、この命令実行処理及びオブジェクトチェーン追跡処理によって、GCフラグ325がONである場合において、オブジェクトポインタが処理対象とされたときにはそのオブジェクトポインタから辿ることのできるオブジェクトポインタはいずれもオブジェクト参照情報のメモリ領域に格納されることになる。

[0063]

なお、スレッドによるオブジェクトポインタを処理対象とする演算の実行によって、そのオブジェクトポインタで指されていたオブジェクトにそのスレッドからアクセスすることが不可能な状態となった場合、つまり、スレッドの動作により、(a) オブジェクトポインタがクリアされた場合、(b) オブジェクトポインタが他の内容に更新された場合、或いは、(c) オブジェクトポインタがスタック領域の有効範囲内に存在する場合においてスタックポインタの変更がなされてスタック領域の有効範囲外となりスレッドから基本的にアクセス不能となった場合にも、そのスレッドによって予めそのオブジェクトポインタが他のスレッド



<実施形態2>

0

以下、本発明の実施形態 2 に係るGCシステムについて図面を用いて説明する

[0064]

実施形態2に係るGCシステムは、APのスレッドを順次停止して参照処理を行う点で、実施形態1に係るGCシステム10と基本的に同様である。ただし、実施形態2に係るGCシステムは、主に、スレッド毎にオブジェクト参照情報やGCフラグを有するのではなくシステム全体でまとめてオブジェクト参照情報やGCフラグを有する点や、スレッド情報の内容を一般的なマルチスレッド制御に要する程度に抑えている点が、GCシステム10と異なる。

[0065]

図15は、本発明の実施形態2に係るGCシステムの機能ブロック図である。 図15に示すGCシステム20の構成要素のうち、実施形態1で示したGCシステム10と同様の構成要素については、図1と同じ符号を付しており、ここではGCシステム20に特有の点を中心に説明する。なお、特に説明しない点については、GCシステム20はGCシステム10と同様である。

[0066]

GCシステム20は、図15に示すように、インタプリタ部1100、オブジェクト管理部200、スレッド管理部1300及びGC部1400を備える。

ここで、インタプリタ部 1 1 0 0 は、基本的にインタプリタであって A P を実行する機能を担い、命令実行部 1 1 0 及びオブジェクト参照検出部 1 1 2 0 を有する。

[0067]

スレッド管理部1300は、マルチスレッド制御を行いスレッドを管理する機能を担い、スレッド制御部310と、スレッド毎に、スレッド情報1320及びスタック340に対応するメモリ領域を有する。なお、スレッド情報1320は



、図16に示すように状態321、優先度322及びスタックポインタ323を含むものである。

[0068]

GC部1400は、基本的にいわゆるガーベジコレクタに相当し、GC制御部1410と、スレッド選定条件記憶部420と、参照処理部1430と、解放部440と、オブジェクト参照情報1450及びGCフラグ1460に対応するメモリ領域とを有する。

インタプリタ部1100におけるオブジェクト参照検出部120は、GCフラグ1460がONである場合に限って、命令実行部110により実行されるスレッド内の命令が、オブジェクトのオブジェクトポインタを処理対象とする命令であるときに、そのオブジェクトポインタを、オブジェクト参照情報1450のメモリ領域の中に格納する機能を有する。

[0069]

GC部1400を中心として行われるGCは、基本的にマークアンドスイープ方式を用い、参照処理の対象となるAPスレッドを順番に選択して、選択したスレッドを停止させて参照処理を行ってからその停止を解除する方式のものである。

GC部1400における参照処理部1430は、スタック及びオブジェクト参照情報1450を参照することにより参照処理を行う機能を有する。オブジェクト参照情報1450は、GCシステム10におけるオブジェクト参照情報と同様にオブジェクトポインタを内容とする情報であるが、スレッド毎に存在するのではない。このオブジェクト参照情報1450のメモリ領域には、APの全スレッドについてのオブジェクト参照検出部1120により、オブジェクトポインタが命令の処理対象とされた場合にそのオブジェクトポインタが格納される。

[0070]

また、GC制御部1410は、図17に示すGC制御処理を実行する機能を有する。

図17は、実施形態2におけるGC制御処理を示すフローチャートである。なお、このフローチャート中の処理ステップのうち、実施形態1のGC制御処理と



同様のものについては、図8と同じ符号を付して示している。

[0071]

このGC制御処理は、タイマーに基づき一定周期で行われる。

GC制御部1410は、最初に、fromテーブルポインタ211とtoテーブルポインタ212の内容を交換することにより、fromテーブルとtoテーブルとの内容を切り替え(ステップS11)、GCフラグ1460をONにする(ステップS111)。

[0072]

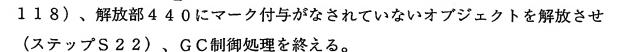
ステップS111に続いて、GC制御部1410は、共有オブジェクト管理情報353内のオブジェクトポインタを辿ることで到達するオブジェクトに対するマーク付与を行うための共有オブジェクト参照処理を行い(ステップS15)、GC未処理スレッドつまり未だ参照処理の対象として選定されていないスレッドが存在するか否かを判定する(ステップS16)。

[0073]

ステップS16においてGC未処理スレッドが存在すると判定した場合には、GC制御部1410は、参照処理の対象となるスレッドを決定するための対象スレッド決定処理を行い(ステップS17)、スレッド制御部310にその決定したスレッドを停止させ(ステップS18)、その決定したスレッドに対応するスタックポインタの位置に着目して(ステップS112)、参照処理部1430に参照処理(図12参照)を実行させ(ステップS113)、スレッド制御部310にその決定したスレッドの停止を解除させて(ステップS21)、再度、ステップS16の判定に戻る。

[0074]

また、ステップS16において、GC未処理スレッドが存在しないと判定した場合には、GC制御部1410は、スレッド制御部310にAPの全スレッドを停止させ(ステップS114)、オブジェクト参照情報1450のメモリ領域の先頭位置に着目し(ステップS115)、参照処理部1430に参照処理を実行させ(ステップS116)、GCフラグ1460をOFFにし(ステップS117)、スレッド制御部310にAPの全スレッドの停止を解除させ(ステップS



[0075]

ここで、インタプリタ部1100によりなされる命令実行処理について簡単に 説明する。

図18は、実施形態2における命令実行処理を示すフローチャートである。

同図に示すように、この命令実行処理は、図13で示した命令実行処理からステップS76を削除したものと基本的に等しい。

[0076]

まず、命令実行部110は、スレッドの現在の実行位置における命令を解釈して実行し(ステップS71)、オブジェクト参照検出部1120は、GCフラグ1460がONであるか否かを判定し(ステップS72)、ONでなければ、命令実行部110が次の命令の解釈実行を行う(ステップS71)。

ステップS 7 2 において、GCフラグ3 2 5 がONであると判定した場合には、オブジェクト参照検出部1120は、ステップS 7 1 において命令実行部110により実行された命令が、オブジェクトポインタを処理対象とする命令であったか否かを判定し(ステップS 7 3)、オブジェクトポインタを処理対象とする命令でなかったならば、命令実行部110が次の命令の解釈実行を行う(ステップS 7 1)。

[0077]

ステップS 7 3 において、オブジェクトポインタを処理対象とする命令であったと判定した場合には、オブジェクト参照検出部 1 1 2 0 はそのオブジェクトポインタと同値のものがオブジェクト参照情報領域に既に格納済みであるか否かを判定し(ステップS 7 4)、格納済みである場合には、命令実行部 1 1 0 が次の命令の解釈実行を行う(ステップS 7 1)。

[0078]

ステップS 7 4 において、そのオブジェクトポインタと同値のものがオブジェクト参照情報のメモリ領域に格納済みでない場合には、オブジェクト参照検出部 1 1 2 0 は、そのオブジェクトポインタを、オブジェクト参照情報中の位置に格



納し(ステップS 7 5)、その後は命令実行部 1 1 0 が次の命令の解釈実行を行う(ステップS 7 1)。

[0079]

このようにしてGC制御部1410は、スレッド選定条件を参照することにより、参照処理の対象にするAPスレッドを1つ選定し、選択したスレッドについて、スレッド制御部310にそのスレッドを停止させてから、参照処理部1430にそのスレッドに対応するスタックに基づいて参照処理を行わせ、その後にスレッド制御部310に選択したスレッドの停止を解除させてから、次のAPスレッドの選択を行うという手順を、未処理のAPスレッドが存在しなくなるまで繰り返し、その後、スレッド制御部310にAPの全スレッドを停止させ、参照処理部1430にオブジェクト参照情報に基づいて参照処理を行わせて、スレッド制御部310にAPの全スレッドの停止を解除させて、解放部440に、マーク付与がなされていないオブジェクトの解放を行わせる。

<補足>

以上、本発明に係るGCシステムについて、実施形態1、2に基づいて説明したが、本発明は、勿論これらの実施形態に限られない。以下、実施形態の変形に関して説明する。

(1) スレッド選定条件は、スレッド状態、スレッド優先度、スタックサイズという3つの条件をこの順に優先適用されることとしたが、条件数や適用順序等は、これに限定されることはなく、また、例えばスレッド優先度が高いものを条件にしたりスタックサイズが大きいものを条件にしたりしてもよい。

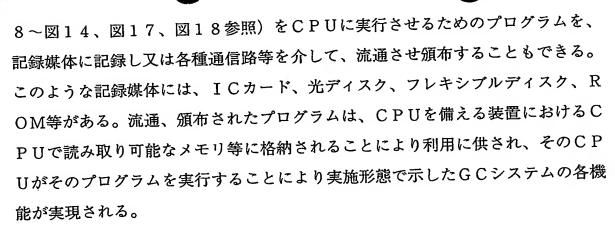
[0080]

但し、実施形態において示したようにスレッド状態がwait状態のものを参照処理対象と選定することは、現在動作しているAPに対する悪影響が少ないという効果につながる。また、参照処理(図12)のアルゴリズムにより、あるスレッドに対する参照処理にかかる時間は、参照処理を先に行うスレッドより後に行うスレッドの方が比較的短くなるため、実施形態において示したようにスレッド優先度が低いものを早めに参照処理対象と選定するようにしておくことで、即応性の求められるところのスレッド優先度の高いスレッドの停止時間を短くし得



るという効果が生じる。また、同様の理由と、スタックサイズが大きいほど参照 処理には時間がかかるという理由とから、対応するスタックサイズの小さいスレッドから順に参照処理を行うようにすることで、同じスレッド優先度のスレッド 間である程度等しくなるようにスレッドの停止時間を分散できるという効果が生 じる。

- (2) 実施形態では、G C制御処理がタイマーに基づき一定周期で行われること としたが、これに限定されることはなく、A P に割り当てられる空きメモリの量 が所定量より少なくなった場合にG C制御処理を行うようにしてもよい。
- (3) 実施形態で示したオブジェクト参照情報のメモリ領域は、連続した物理アドレスで表されるメモリ領域であってもよいし、オブジェクトポインタの格納に際して必要が生じた度に一定量のメモリ領域を追加的に確保してその確保した複数の断続的なメモリ領域を論理アドレス上で連続しているように扱うこととしてもよい。
- (4) 実施形態1で示したGCシステムにおいては、インタプリタによる命令実行処理中に、GCフラグがONであればオブジェクトポインタを処理対象としたときにオブジェクト参照情報のメモリ領域にそのオブジェクトポインタを保存する処理を行っておくこととし、命令実行処理中では、そのオブジェクトポインタと同値のものをfromテーブルからtoテーブルに移行することを内容とする参照処理を行わずインタプリタによる命令実行の高速性を保つようにしたが、命令実行処理中にその参照処理を行うこととしてもよい。
- (5) 実施形態では共有オブジェクト管理情報内のオブジェクトポインタから辿ることのできるオブジェクトへのポインタは参照処理によってfromテーブルからtoテーブルに移行されることとしたが、参照処理を行わなくても、共有オブジェクト管理情報によって管理されているオブジェクトは解放部による解放対象から除外しさえすればよい。
- (6) 実施形態で示したGCシステムは、コンピュータ上に実装されるのみならず、CPUを備える携帯端末、家電機器等において実装されることとしてもよい
 - (7) 実施形態で示したGCシステムの各機能を実現させるための各種処理(図



[0081]

【発明の効果】

以上説明したように、本発明に係るGCシステムは、複数スレッドで構成されるオブジェクト指向プログラムの実行過程において不要になったオブジェクトに対応するメモリ領域を解放するガーベジコレクションシステムであって、複数スレッドそれぞれを順に選択する選択手段と、選択されたスレッドについて、まず当該スレッドの実行を停止して、その状態で、当該スレッドからオブジェクトポインタの参照を通じてアクセス可能であるオブジェクトを検出して当該検出したオブジェクトを非解放対象として管理して、その後に、当該スレッドの実行を再開するという一連の手順からなる検査処理を実施する検査手段と、前記選択手段による前記選択が開始された後において、実行中のスレッドによりオブジェクトポインタが処理対象にされたことを検知した場合には、当該オブジェクトポインタが指すオブジェクトを非解放対象として管理する検知手段と、前記複数スレッド全てについて前記検査処理が完了した後において、非解放対象として管理されているオブジェクト以外のオブジェクトに対応するメモリ領域を、解放する解放手段とを備えることを特徴とする。

[0082]

ここで、オブジェクトを非解放対象として管理するとは、例えばそのオブジェクトポインタをfromテーブルからtoテーブルに移行する上述の方法等により、マーク付与を実現することをいう。

これにより、APスレッドからスタック内やオブジェクト内のオブジェクトポインタを介して辿ることができるオブジェクトについての非解放対象としての指



定の過程つまりマーク付与の過程においては、そのAPスレッドを停止させているため、そのAPスレッドが動作することに起因してスタック領域内のデータが変化するという事態が起こらないため、その過程におけるマーク付与が無駄になる危険性はなくなり、この点についてはCPUの利用効率低下を防止できる。

[0083]

また、これにより、全てのAPスレッドを停止して上述のマーク付与を行っているのではないため、APの全スレッドが停止している期間の長期化が防止できる。なお、全APスレッドを停止させないことによるオブジェクトへの参照状態の変化に対しては、実行中のスレッドによりオブジェクトポインタが処理対象とされたことを監視して対応しているため、いずれかのスレッドからアクセス可能なオブジェクトについては必ず非解放対象として管理されることになる。

[0084]

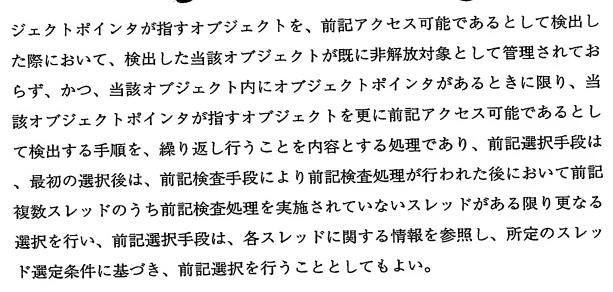
また、前記検知手段は、前記検知を、実行中のスレッドについて未だ検査処理がなされていない場合に限り行い、前記検知手段は、前記検知をした場合に、当該実行中のスレッドに処理対象にされたオブジェクトポインタ、及び当該オブジェクトポインタから辿ることのできるオブジェクト内のオブジェクトポインタを、当該スレッドに対応する作業用メモリ領域に保存する検出部と、前記検査手段によりスレッドの実行が停止されている間に、当該スレッドに対応する作業用メモリ領域内のオブジェクトポインタから辿ることのできるオブジェクトを非解放対象として管理する管理部とを有することとしてもよい。

[0085]

これにより、スレッドが、参照処理つまり参照されているオブジェクトへのマーク付与の処理の対象とされるまでの間だけ、インタプリタによるそのスレッドの実行時にオブジェクトポインタが処理対象とされたことを検出して作業用メモリ領域、即ちオブジェクト参照情報のメモリ領域に格納する処理を行うため、一旦スレッドが参照処理の対象とされた後は、そのスレッドはインタプリタによる実行時に、その検出がない分だけ、より高速に動作するようになる。

[0086]

また、前記検査処理は、選択された当該スレッドに対応するスタック内のオブ



[0087]

これにより、既に非管理対象として管理されたオブジェクトの配下のオブジェクトを重複的に検出のための処理の対象とすることを防止することができる。また、この重複的に検出のための処理を行わない仕組みにより、参照処理の対象として早期に選択されたスレッドより、遅く選択されたスレッドの方が参照処理に係る時間が短縮される可能性が高まるので、各スレッドに要求される応答性能に鑑みて、予めスレッド選定条件を定めておくことで、各スレッドが適切な応答性能を発揮して実行されるようにある程度制御することができるようになる。

[0088]

また、前記スレッド選定条件は、スレッド状態がwait状態であるスレッドをスレッド状態がwait状態以外であるスレッドよりも早期に選択することを示す条件を含み、前記選択手段は、前記選択を行う時点でwait状態のスレッドがある限り、当該wait状態のスレッドを選択することとしてもよい。

これにより、wait状態のスレッドを停止することになるため、現在動作しているAPスレッドに対する悪影響の発生が抑えられる。

[0089]

また、前記スレッド選定条件は、スレッド優先度のスレッド優先度の低いスレッドをスレッド優先度の高いスレッドよりも早期に選択することを示す条件を含むこととしてもよい。

これにより、スレッド優先度の高いスレッドは、参照処理が短時間でなされる



可能性が高くなり、実行性能の低下がある程度防止されるようになる。

[0090]

また、前記スレッド選定条件は、スレッドに対応するスタックサイズが小さい スレッドをスタックサイズが大きいスレッドよりも早期に選択することを示す条 件を含むこととしてもよい。

これにより、スレッドからアクセス可能なオブジェクトへのオブジェクトポインタを格納するスタックの有効範囲が小さい程、参照処理に要する時間は短くなる傾向にあることに鑑みれば、各スレッドの停止時間をある程度均等化する効果が得られ、特定のスレッドの応答性が他と比べて際立って悪くなるという事態がある程度回避できるようになる。

【図面の簡単な説明】

【図1】

本発明の実施形態1に係るGCシステムの機能ブロック図である。

【図2】

オブジェクトとオブジェクトポインタとの関係を例示する図である。

【図3】

fromテーブル及びtoテーブルを示す図である。

【図4】

参照処理によるfromテーブル及びtoテーブルの内容の変化を示す図である。

【図5】

スレッド情報、スタック及びオブジェクト参照情報を示す図である。

【図6】

参照処理とAPスレッドの状態との関係を示す図である。

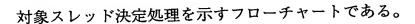
【図7】

スレッド選定条件の内容を示す図である。

【図8】

GC制御処理を示すフローチャートである。

【図9】



【図10】

共有オブジェクト参照処理を示すフローチャートである。

【図11】

対象スレッド参照処理を示すフローチャートである。

【図12】

参照処理を示すフローチャートである。

【図13】

命令実行処理を示すフローチャートである。

【図14】

オブジェクトチェーン追跡処理を示すフローチャートである。

【図15】

本発明の実施形態2に係るGCシステムの機能ブロック図である。

【図16】

スレッド情報及びスタックを示す図である。

【図17】

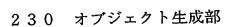
実施形態2におけるGC制御処理を示すフローチャートである。

【図18】

実施形態2における命令実行処理を示すフローチャートである。

【符号の説明】

- 100、1100 インタプリタ部
- 110 命令実行部
- 120、1120 オブジェクト参照検出部
- 200 オブジェクト管理部
- 210 オブジェクト情報記憶部
- 211 fromテーブルポインタ
- 212 toテーブルポインタ
- 221 fromテーブル
- 222 toテーブル



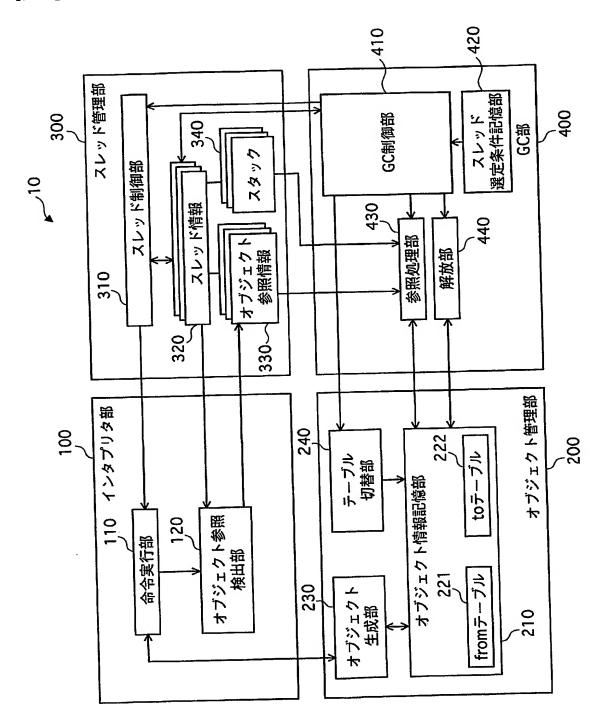
- 2 4 0 テーブル切替部
- 300、1300 スレッド管理部
- 3 1 0 スレッド制御部
- 320、1320 スレッド情報
- 330 オブジェクト参照情報
- 340 スタック
- 400、1400 GC部
- 410、1410 GC制御部
- 420 スレッド選定条件記憶部
- 430、1430 参照処理部
- 4 4 0 解放部
- 1450 オブジェクト参照情報



【書類名】

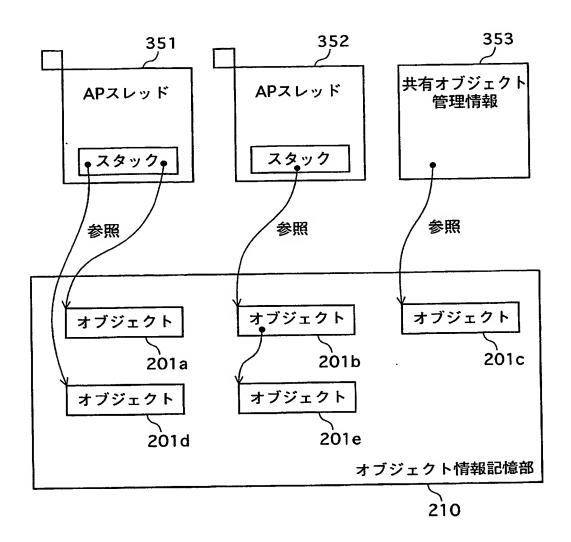
図面

【図1】



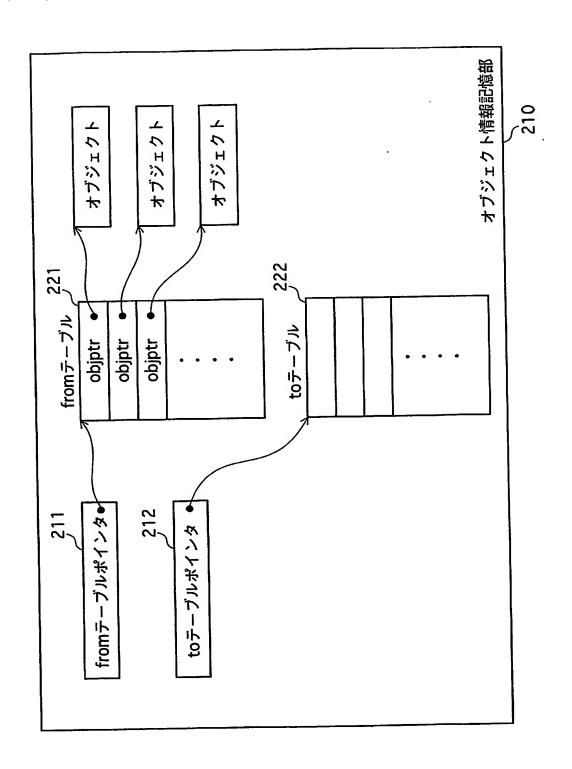


【図2】



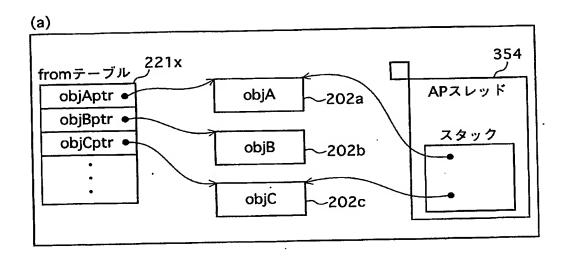


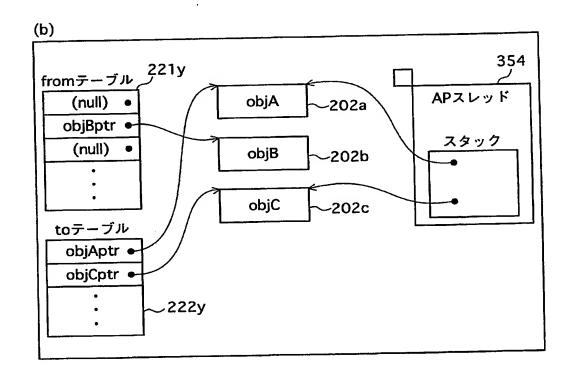
【図3】





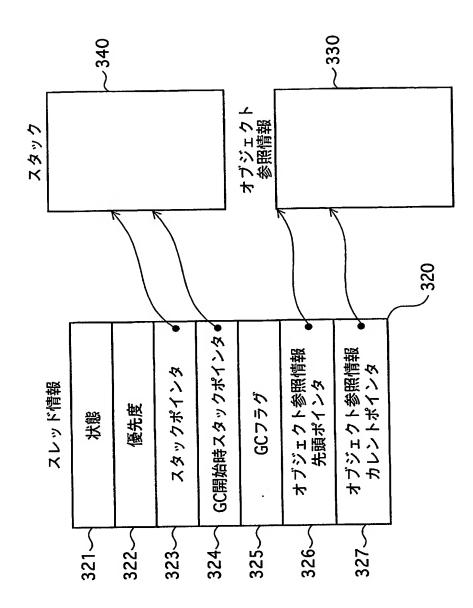
【図4】





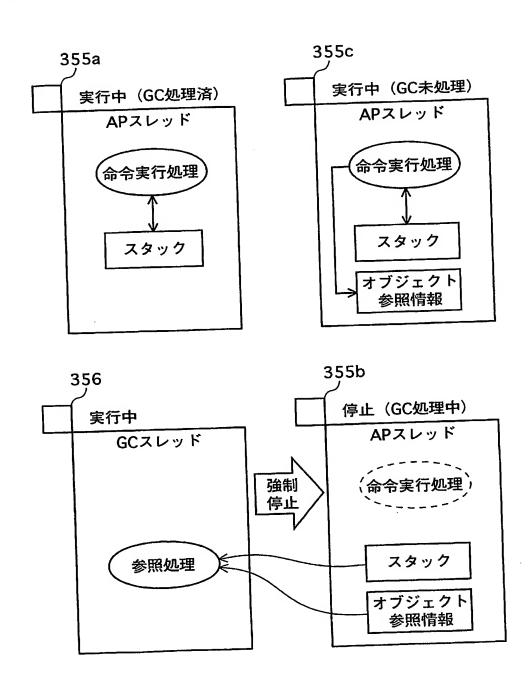


【図5】





【図6】





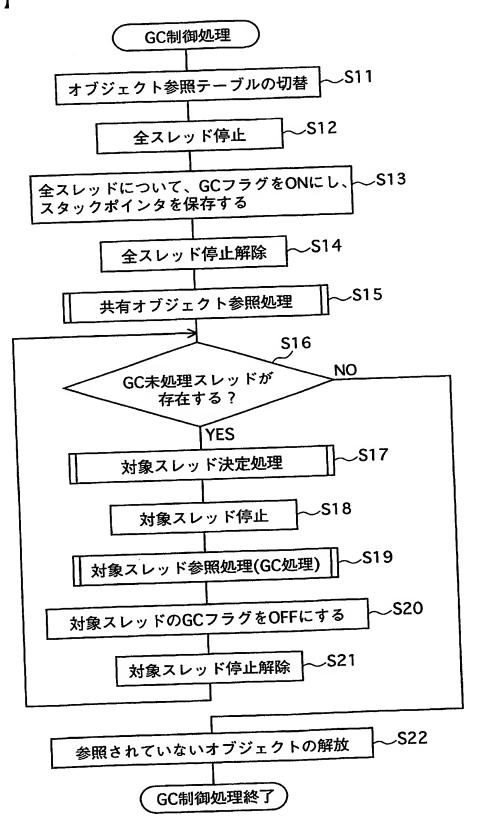
【図7】

スレッド選定条件

422~	スレッド状態	WAIT 状態	
423 ~	スレッド優先度	低い	
424 ~	スタックサイズ	小さい	
			421

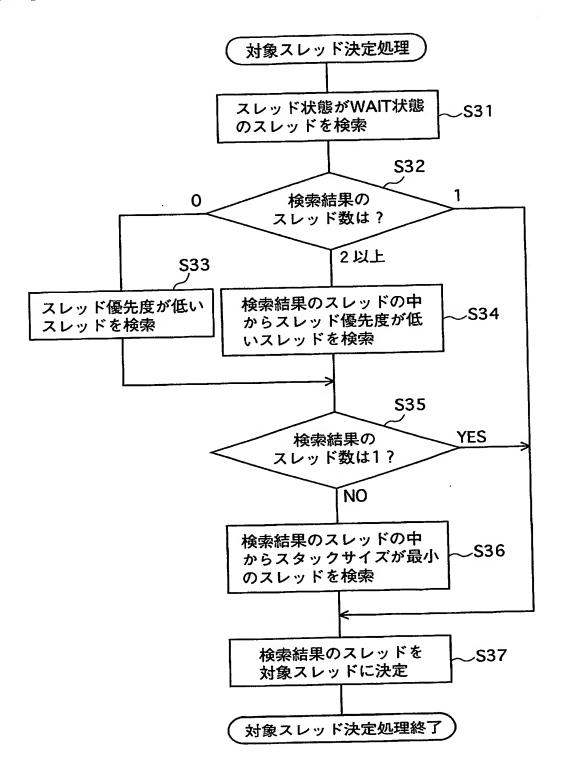


【図8】



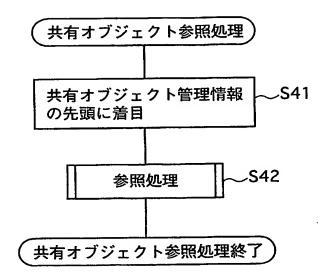


【図9】



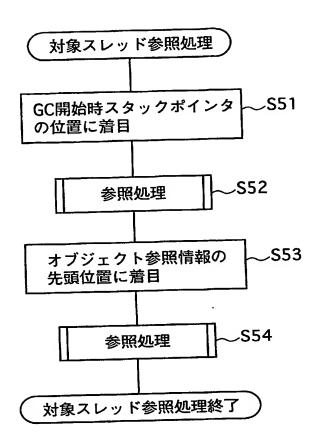


【図10】



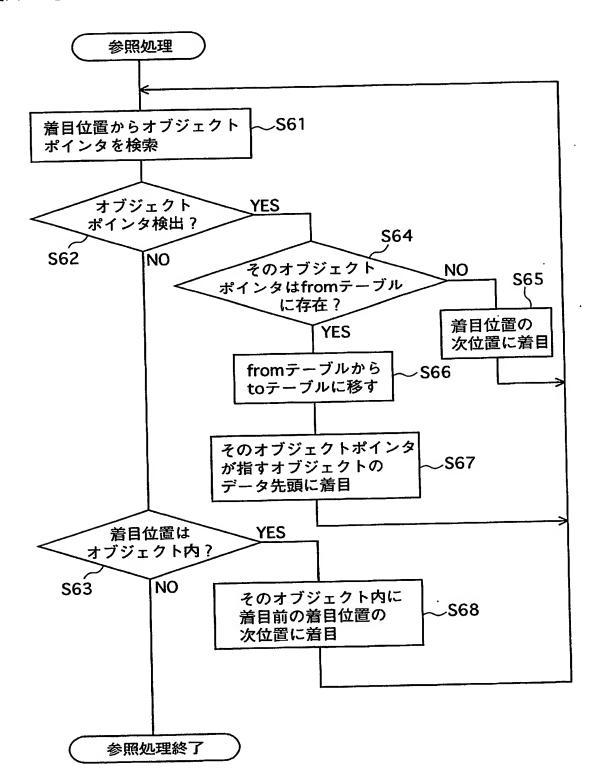


【図11】



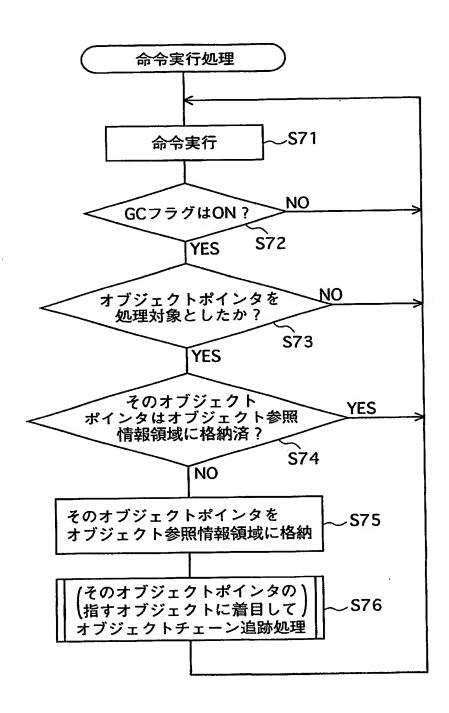


【図12】



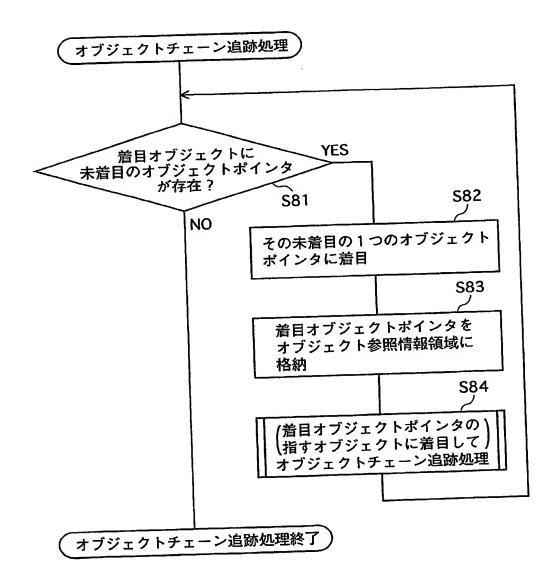


【図13】



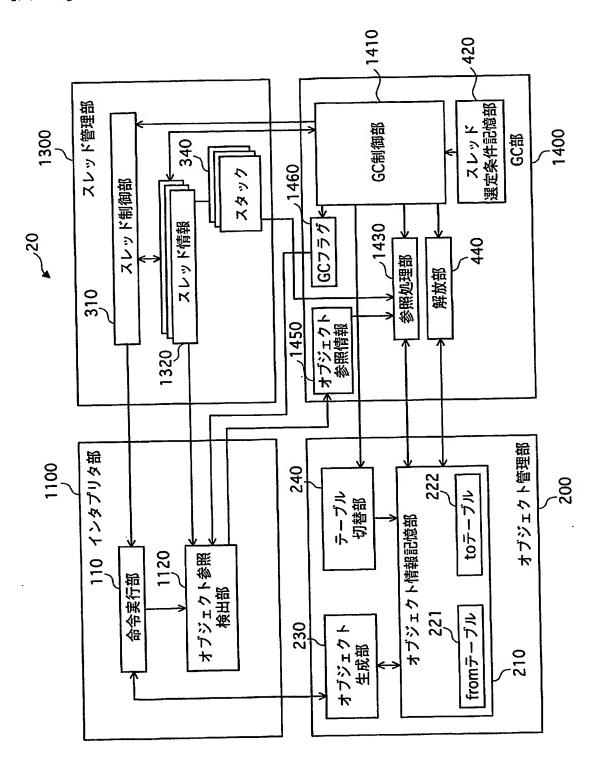


【図14】



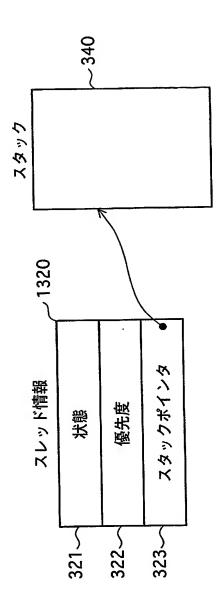


【図15】

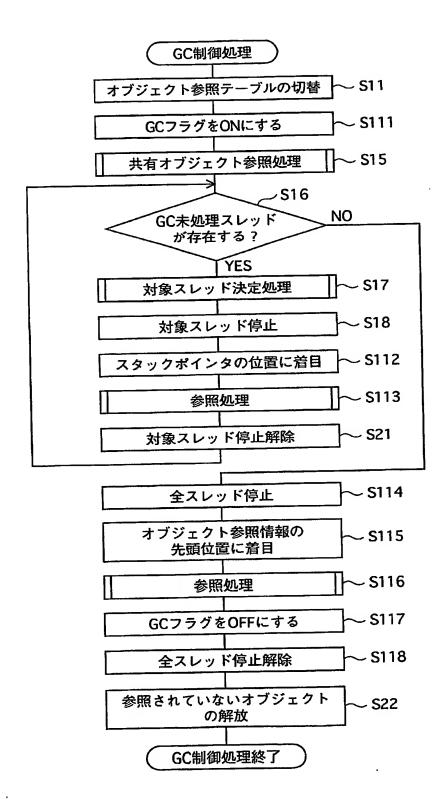




【図16】

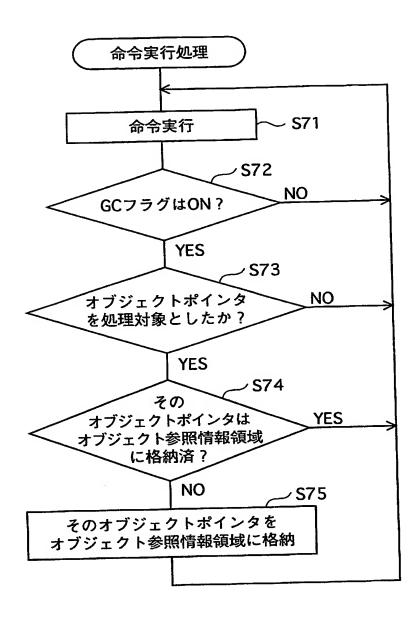








【図18】





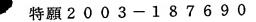
【書類名】 要約書

【要約】

【課題】 APの全スレッド停止の期間を長期化させず、GCに要するCPU時間の無駄な増加をある程度抑制するガーベジコレクション(GC)システムを提供する。

【解決手段】 GC制御部410は、スレッド選定条件記憶部420の内容に従って各スレッドを順次選択し、選択したスレッドについて、実行停止し、参照処理部430により、そのスレッドからオブジェクトポインタの参照を通じてアクセス可能なオブジェクトを検出してそのオブジェクトを非解放対象として管理する参照処理を実行して、そのスレッドの実行を再開する。また、インタプリタ部100は、実行中スレッドによりオブジェクトポインタが処理対象にされたことを検知すればそのオブジェクトポインタを記録する。参照処理では更にその記録されたオブジェクトポインタから辿れるオブジェクトも非解放対象として管理する。解放部440は非解放対象以外のオブジェクトを解放する。

【選択図】 図1



出 願 人 履 歴 情 報

識別番号

[000005821]

1. 変更年月日

1990年 8月28日

[変更理由]

新規登録

住 所 氏 名 大阪府門真市大字門真1006番地

松下電器産業株式会社